On Benefits of Equality Constraints in Lex-Least Invariant CAD (Extended Abstract)

Akshar Nair[1 [0000−0001−7379−1868]], James Davenport[1 [0000−0002−3982−7545]], and Gregory Sankaran[1 [0000−0002−5846−6490]]

University of Bath, Bath, United Kingdom, BA2 7AY.
{asn42,masjhd,masgks}@bath.ac.uk[1]

Abstract. There are two relevant methods for CAD: McCallum [1984] [6] which used order invariant CAD's and Lazard [Lazard1994, McCallumetal2019] [5,10] which used lex-least invariant CADs, and doesn't have the nullification problem of McCallum [1984]. McCallum [1999] [7] was the first to prove a CAD operator based on McCallum [1984], that took advantage of an equational constraint. In this paper, we do the same for Lazard's method. This takes in a lex-least invariant CAD of $\mathbb{R}^{n-1}$ as input and outputs a sign invariant CAD of $\mathbb{R}^n$: consequently, it cannot be used recursively, but only for x_n, the first variable to be projected. In the further steps of the projection phase, we use Lazard's original projection operator. Nonetheless, reducing the output in the first step has a domino effect throughout the remaining steps, which significantly reduces the complexity. The long-term goal is to find a general projection operator that takes advantage of the equality constraint and can be used recursively, and this operator gives an important first step in that direction.

Keywords: Equality Constraint · Cylindrical Algebraic Decomposition · Lex-Least invariance.

1 Introduction

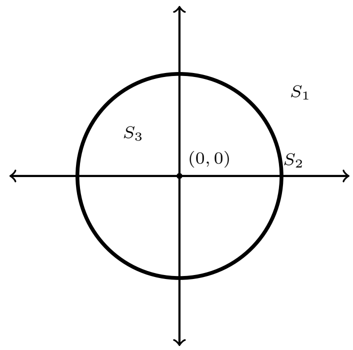
A Cylindrical Algebraic Decomposition (CAD) is a decomposition of a semi-algebraic set $S \subseteq \mathbb{R}^n$ (for any n) into semi-algebraic sets (also known as cells) homeomorphic to $\mathbb{R}^m$, where $1 \le m \le n$, such that the projection of any two cells onto the first k coordinates is either the same or disjoint. We generally want the cells to have some property relative to some given set of input polynomials. For example, we might require sign-invariance, i.e. the sign of each input polynomial (often referred to as a constraint) is constant on each cell. Many algorithms for CAD consist of the following three phases.

Projection phase: This phase consists of reducing the number of variables of the polynomial constraints (using a function known as the projection operator) until it has reached polynomials in one variable ($\mathbb{R}[x_1]$). The variable ordering is important (generally $x_1 > x_2 > \ldots > x_n$) i.e. the first variable eliminated is x_n.

Base phase: Decompose $\mathbb{R}^1$ according to specifications of the required CAD. Each cell is given a sample point, which is used for tracking cells in the lifting phase.

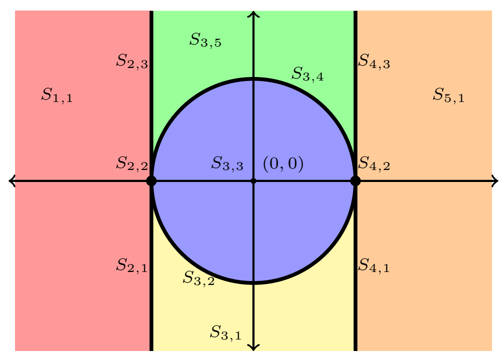
Lifting phase: This phase consists of decomposing higher dimensional spaces using the sample points and the projection polynomials in that dimension, until S is decomposed.

Example 1. Let us look at the decomposition of $S = \mathbb{R}^2$ with respect to the input polynomial: $x^2 + y^2 - 1$.



$$S_1 = \{x^2 + y^2 - 1 > 0\}; S_2 = \{x^2 + y^2 - 1 = 0\}; S_3 = \{x^2 + y^2 - 1 < 0\}$$

This is not a cylindrical decomposition of $\mathbb{R}^2$, because we project S_1 and S_2 to $\mathbb{R}^1$ we get $\mathbb{R}$ and $(-1, 1)$ respectively. These have a non-empty intersection but are not the same. To decompose it cylindrically (with ordering $x > y$), we take:



$$\begin{aligned}
&S_{1,1} = \{x < -1\}, \quad S_{5,1} = \{x > 1\}, \\
&S_{2,1} = \{x = -1, y < 0\}, \quad S_{2,2} = \{x = -1, y = 0\}, \quad S_{2,3} = \{x = -1, y > 0\}, \\
&S_{3,1} = \{x^2 + y^2 - 1 > 0 \land 1 > x > -1 \land y < 0\}, \quad S_{3,2} = \{x^2 + y^2 - 1 = 0 \land 1 > x > -1 \land y < 0\}, \\
&S_{3,3} = \{x^2 + y^2 - 1 < 0 \land 1 > x > -1\}, \quad S_{3,4} = \{x^2 + y^2 - 1 = 0 \land 1 > x > -1 \land y > 0\}, \\
&S_{3,5} = \{x^2 + y^2 - 1 > 0 \land 1 > x > -1 \land y > 0\}, \\
&S_{4,1} = \{x = 1, y < 0\}, \quad S_{4,2} = \{x = 1, y = 0\}, \quad S_{4,3} = \{x = 1, y > 0\}.
\end{aligned}$$

The first CAD algorithm was given by Collins in 1975 [2], which has complexity $(2dm)^{3^{2n+O(1)}}$ when the input consists of m polynomials of at most degree d in n variables [3]. In particular the algorithm of [2] lifted sign-invariant CADs. CAD algorithms have many applications: epidemic modelling [1], artificial intelligence to pass exams [12], financial analysis [11], and many more. If S is contained within a subvariety it is clearly wasteful to compute a decomposition of $\mathbb{R}^n$. To make this idea more precise, we need some terminology.

Definition 1. *A Quantifier Free Tarski Formula (QFF) is made up of atoms connected by the standard boolean operators* $\wedge, \vee$ *and* $\neg$. *The atoms are statements about signs of polynomials* $f \in \mathbb{Z}[x_1, \ldots, xn]$*:* $f \ \rho \ 0$ *where* $\rho \in \{=, <, >\}$ *(and by combination also* $\{\geq, \leq, \neq\}$*).*

Strictly speaking we need only the relation $<$, but this form is more convenient because of the next definition.

Definition 2. *[4] An Equational Constraint (EC) is a polynomial equation logically implied by a QFF. If it is an atom of the formula, it is said to be* explicit*; if not, then it is* implicit*. If the constraint is visibly an equality one from the formula, i.e. the formula* Φ *is* $f = 0 \wedge \Phi'$*, we say the constraint is* syntactically explicit*.*

Although implicit and explicit ECs have the same logical status, in practice only the syntactically explicit ECs will be known to us and therefore be available to be exploited.

Example 2. [4] Let f and g be two polynomials,

1. The formula $f = 0 \wedge g > 0$ has an explicit EC $f = 0$.
2. The formula $f = 0 \vee g = 0$ has no explicit EC, however the equation $fg = 0$ is an implicit EC.
3. The formula $f^2 + g^2 \leq 0$ also has no explicit EC, but it has two implicit EC: $f = 0$ and $g = 0$.
4. The formula $f = 0 \vee f^2 + g^2 \leq 0$ logically implies $f = 0$, and the equation is an atom of the formula which makes it an explicit EC according to the definition, Since this deduction is semantic rather than syntactic, it is more like an implicit EC rather than an explicit EC

Definition 3. *Let* A *be a set of polynomials in* $\mathbb{R}[x_1, \ldots, x_n]$ *and* $P\colon \mathbb{R}[x_1, \ldots, x_n] \times \mathbb{R}^n \to \Sigma$ *a function to some set* Σ. *If* $C \subset \mathbb{R}^n$ *is a cell and* $P(f, \alpha)$ *is independent of* $\alpha \in C$ *for every* $f \in A$*, then* A *is called* P*-invariant over that cell. If this is true for all the cells of a decomposition, we say the decomposition is* P*-invariant.*

This idea was first used by McCallum in 1999 [7]: if the QFF that defines S contains an EC then [7] gives an algorithm to perform a decomposition of the varieties described by the EC rather than a decomposition of $\mathbb{R}^n$. This method extends an order invariant CAD of $\mathbb{R}^{n-1}$ to a sign invariant CAD $\mathbb{R}^n$ of some variety. This is not a recursive algorithm, as the output is a sign-invariant CAD: for it to be recursive we would need it to be order-invariant. McCallum modified his result in 2001 [8] so that it would output an order-invariant CAD; this could then be used to deal with multiple equational constraints, reducing the complexity significantly in specific cases.

In 1994, Lazard [5] provided a different approach to constructing CADs: rather than using the order of polynomials, he proposed a valuation linked to their power series expansion at the point in question. This allowed a slightly smaller projection operator but required a significantly different lifting process.

In this paper, we modify Lazard's process for the single equational constraint situation. The result is similar to the projection operator proposed by McCallum in 1999 [7]. It works well when the quantified formula is of the form $(q = 0) \wedge \phi$, but will not work otherwise: in that case, we would resort to using the normal projection operator. As in [7], it cannot be used inductively, since it outputs a sign-invariant CAD rather than a lex-least valuation-invariant CAD.

Our projection operator is based on Lazard's projection which does not ask for middle coefficients as compared to McCallum's projection operator. Our improvement is that we only need to compute Lazards projection operator on the equational constraints. The only additional polynomials are the resultants between the equational and non-equational constraints. This is a considerable improvement over Collins' projection operator and McCallum's modifications, as the projection set in our case is significantly smaller. This will be further discussed in Section 4.

2 Lex-least Valuation

In order to understand lex-least valuation, let us recall *lexicographic order* $\geq_{lex}$ on $\mathbb{N}^n$, where $n \geq 1$. We say that $v = (v_1, \ldots, v_n) \geq_{lex} (w_1, \ldots, w_n) = w$ if and only if either $v = w$ or there exists an $i \leq n$ such that $v_i > w_i$ and $v_k = w_k$ for all k in the range $1 \leq k < i$.

Definition 4. *[10, definition 2.4] Let $n \geq 1$, $f \in \mathbb{R}[x_1, ..., x_n]$ non-zero and $\alpha = (\alpha_1, ..., \alpha_n) \in \mathbb{R}^n$. The lex-least valuation $\nu_\alpha(f)$ at α is the least (with respect to $\geq_{lex}$) element $v = (v_1, ..., v_n) \in \mathbb{N}^n$ such that f expanded about α has the term*

$$c(x_1 - \alpha_1)^{v_1} \cdots (x_n - \alpha_n)^{v_n},$$

where $c \neq 0$.

Note that $\nu_\alpha(f) = (0, \ldots, 0)$ if and only if $f(\alpha) \neq 0$. The lex-least valuation is referred to as the Lazard valuation in [10].

Example 3. Let $n = 1$ and $f(x_1) = x_1^3 - 2x_1^2 + x_1$, then we have $\nu_0(f) = 1$ and $\nu_1(f) = 2$. Let $n = 2$ and $f(x_1, x_2) = x_1(x_2 - 1)^2$. Then we get $\nu_{(0,0)}(f) = (1, 0)$, $\nu_{(2,1)}(f) = (0, 2)$ and $\nu_{(0,1)}(f) = (1, 2)$.

The following are the important properties of the lex-least valuation. These are mostly taken from [10]. In some cases where we wish to draw attention to the details of the proof, we have outlined them below.

Proposition 1. *[10, proposition 3.1](Valuation) ν_α is a valuation, i.e. if f and g are non-zero elements of $\mathbb{R}[x_1, \ldots, x_n]$ and $\alpha \in \mathbb{R}^n$, then $\nu_\alpha(fg) = \nu_\alpha(f) + \nu_\alpha(g)$ and $\nu_\alpha(f+g) \geq_{lex} \min\{\nu_\alpha(f), \nu_\alpha(g)\}$.*

Proposition 2. *[10, proposition 3.2](Upper semicontinuity) Let f be a nonzero element of $\mathbb{R}[x_1, \ldots, x_n]$ and let $a \in \mathbb{R}^n$. Then there exists a open neighbourhood $V \subset \mathbb{R}^n$ of a, such that $\nu_b(f) \leq_{lex} \nu_a(f)$ for all $b \in V$.*

Proposition 3. *[10, proposition 3.4] Let f, g be non-zero elements of $\mathbb{R}[x_1, \ldots, x_n]$ and let $S \subset \mathbb{R}^n$ be connected. Then fg is lex-least invariant in S if and only if f and g are lex-least invariant in S.*

Proof. The reverse implication is straightforward. Now let us assume fg is lex-least invariant in S. Say at a point $\alpha \in S$ the lex-least valuation $\nu_\alpha(f)$ jumps up, this would imply that the lex-least valuation $\nu_\alpha(g)$ jumps down as a consequence of Proposition 1. This would contradict the upper-semicontinuity of the lex-least valuation for g.

Definition 5. *[10] Let $n \geq 2$, take a non-zero element f in $\mathbb{R}[x_1, ..., x_n]$ and let $\beta \in \mathbb{R}^{n-1}$. The Lazard residue $f_\beta \in \mathbb{R}[x_n]$ of f at β, and the lex-least valuation of f at β are defined to be the result of Algorithm 1:*

The value of $(\nu_1, \ldots, \nu_{n-1})$ is called the lex-least valuation of f above $\beta \in \mathbb{R}^{n-1}$ (not to be confused with the lex-least at $\alpha \in \mathbb{R}^n$ definition 4).

Definition 6. *[10, definition 2.10] Let $S \subseteq \mathbb{R}^{n-1}$ and $f \in \mathbb{R}[x_1, \ldots, x_n]$. We say that f is Lazard delineable on S if:*

i) The lex-least valuation of f above β is the same for each point $\beta \in S$.

Algorithm 1 *Lazard residue*

1: $f_\beta \leftarrow f$
2: **for** $i \leftarrow 1$ to $n-1$ **do**
3: $\nu_i \leftarrow$ greatest integer ν such that $(x_i - \beta_i)^\nu | f_\beta$.
4: $f_\beta \leftarrow f_\beta / (x_i - \beta_i)^{\nu_i}$.
5: $f_\beta \leftarrow f_\beta(\beta_i, x_{i+1}, ..., x_n)$
6: **end for**
7: **return** f_β, $(\nu_1, \ldots, \nu_{n-1})$

end

ii) There exist finitely many continuous functions $\theta_1 < ... < \theta_k$ from $S \to \mathbb{R}$ with $k \geq 0$ such that for all $\beta \in S$, the set of real roots of f_β is $\{\theta_1(\beta), ..., \theta_k(\beta)\}$.
iii) If $k = 0$, then the graph of f does not pass over S. If $k \geq 1$, then there exist positive integers $m_1, \ldots, m_k$ such that, for all $\beta \in S$ and for all $1 \leq i \leq k$, m_i is the multiplicity of $\theta_i(\beta)$ as a root of f_β.

Definition 7. *[10, Definition 2.10] Let f be Lazard delineable on $S \subseteq \mathbb{R}^{n-1}$. Then*

i) The graphs θ_i are called Lazard sections and m_i is the associated multiplicity of these sections.
ii) The regions between consecutive Lazard sections[1] are called Lazard sectors.

Remark 1. There is a problem when θ_i's are vertical so we introduce some geometric terminology.

Definition 8. *A variety $C \subseteq \mathbb{R}^n$ is called a curtain if, whenever there exists $(\underline{x}, x_n) \in C$, then $(x, y) \in C$ for all $y \in \mathbb{R}$.*

Definition 9. *Suppose $f \in \mathbb{R}[x_1, \ldots, x_n]$ and $W \subseteq \mathbb{R}^{n-1}$. We say that f has a curtain at W if for all $\underline{x} \in W$ and $y \in \mathbb{R}$ we have $f(\underline{x}, y) = 0$.*

Remark 2. Lazard delineability is very similar to delineability as in [2] and [7], but with two important differences. First, f needs to be lex-least invariant on the section. Second,delineability is not defined on curtains, but Lazard delineability is because the Lazard sections are of f_β rather than f.

Proposition 4. *[10, proposition 2.11] Let $f \in \mathbb{R}[x_1, \ldots, x_n] \backslash \mathbb{R}[x_1, \ldots, x_{n-1}]$ and let S be a connected subset of $\mathbb{R}^{n-1}$. Suppose that f is Lazard delineable on S. Then f is lex-least invariant in each section and sector of f over S.*

Proof. We know that the lex-least valuation of f above β, for all $\beta \in S$, is invariant, meaning that for α in the sections of f over S, the first $n-1$ coordinates of the $\nu_\alpha(f)$ is equal to $\nu_\beta(f)$. Consider a section θ; by the definition of Lazard delineability we know that there is an associated multiplicity m. This means that the last coordinate of the valuation of f above β is m. Hence f is lex-least invariant in every section of f over S. The valuation of f at each sector over S is just $(0, \ldots, 0)$.

Remark 3. We can use Algorithm 1 to compute the lex-least valuation of f at $\alpha \in \mathbb{R}^n$. After the loop is finished, we proceed to the first step of the loop and perform it for $i = n$ and the n-tuple $\nu_1, \ldots, \nu_n$ is the required valuation.

[1] Including $\theta_0 = -\infty$ and $\theta_{k+1} = +\infty$.

Lemma 1. *Let f be an analytic function in r variables such that $f(0,\ldots,0,x_r) = cx_r^m$, with c non-zero. Then $\upsilon_\alpha(f) = (0,\ldots,0,m)$ where $\alpha = (0,\ldots,0)$.*

3 Lex-least Invariance

This section contains details of Lazard's original projection operator used to obtain a lex-least invariant CAD.

Definition 10. *[10, definition 2.1] Let A be a finite set of irreducible polynomials in $\mathbb{R}[x_1,...,x_n]$ with $n \geq 2$. The Lazard projection $PL(A)$ is a subset of R_{n-1} composed of the following polynomials.*

i) All leading coefficients of the elements of A.
ii) All trailing coefficients of the elements of A.
iii) All discriminants of the elements of A.
iv) All resultants of pairs of distinct elements of A.

Experimentally this projection operator gives a more efficient CAD, in practice and on average [9]. This projection set is fairly large in the event that we have a large number of polynomials in A. On the other hand, it is smaller than McCallum's projection set in [6], as it only asks for the leading and trailing coefficients, compared to asking for all of them. McCallum et al. [10] gave a proof for the following theorem. This sets the foundation for the use of Lazard's projection operator in producing a lex-least invariant CAD.

Theorem 1. *[10, Theorem 5.1] Let $f \in \mathbb{R}[x, x_n]$, where $\underline{x} = (x_1,...,x_{n-1})$ with positive degree d in x_n. Let D, l and t denote the discriminant, leading coefficient and trailing coefficient (i.e. the coefficient independent of x_n) of f respectively, and suppose that each of these polynomials are not identically zero (as elements of $\mathbb{R}[\underline{x}]$). Let S be a connected submanifold of $\mathbb{R}^{n-1}$ in which D, l and t are all lex-least invariant. Then f is Lazard delineable on S, and this implies that f is lex-least invariant in every Lazard section and sector over S.*

In summary, Theorem 1 implies that if $PL(A)$ is lex-least invariant over a section S, then all polynomials in A are Lazard delineable over S. This implies that the projection operator can be used inductively, and this makes it possible to provide a lex-least invariant CAD.

4 Modification to Lazard's Projection Operator

This section focuses on the main claim of this paper. We first define our modification of Lazard's projection operator. This projection operator reduces the projection set in the first phase, where there is an EC in the QFF. This is similar to McCallum's modification in 1999 [7]. The reason for considering the resultants here is the same; we are performing the CAD on the hypersurface $f = 0$, rather than $\mathbb{R}^n$.

Definition 11. *Let A be a set of polynomial constraints. Let $E \subseteq A$, and define the projection operator $PL_E(A)$ as follows:*

$$PL_E(A) = PL(E) \cup \{\mathrm{res}_{x_r}(f, g) \mid f \in E, g \in A \setminus E\}.$$

Remark 4. Note that in theory, E could consist of any polynomial but in practice, it needs to be the equational constraints in order to benefit from them.

Theorem 2. *Let $r \geq 2$ and let $f, g \in \mathbb{R}[x_1, \ldots, x_r]$ be real polynomials of positive degrees in the main variable x_r. Let S be a connected subset of $\mathbb{R}^{r-1}$. Suppose that f is Lazard delineable on S, in which $R = \mathrm{res}_{x_r}(f, g)$ is lex-least invariant and that f does not have a curtain on S. Then g is sign-invariant in each section of f over S.*

Proof. Let σ be a section of f over S, given by the map $\theta\colon S \to \mathbb{R}$. Since g is a continuous function it is enough to show that g is sign-invariant in a euclidean neighbourhood of an arbitrary point $\alpha = (\alpha_1, ..., \alpha_r) \in \sigma$.

If $g(\alpha) \neq 0$, then by continuity it is sign-invariant in a euclidean neighbourhood of α. This in turn implies that g is sign-invariant over σ. So for the remainder of the proof, let us assume that $g(\alpha) = 0$.

Without loss of generality, we can assume that α is the origin. By the definition of θ, we can say that $\theta(0, \ldots, 0) = 0$. By the statement of the theorem, we know that f is Lazard delineable over σ and R is lex-least invariant over σ.

Let $x_r = 0$ be a root of $f(0, \ldots, 0, x_r) = 0$ of multiplicity $m \neq 0$. This implies that $f(0, \ldots, 0, x_r) = \bar{q}(x_r) \cdot x_r^m$ where $\bar{q}(0) \neq 0$. Therefore, by Hensel's Lemma [7], there exists a euclidean neighbourhood N_1 about the origin with formal power series $q(x_1, \ldots, x_r)$ and $h(x_1, \ldots, x_r)$ such that $f(x_1, \ldots, x_r) = q(x_1, \ldots, x_r) \cdot h(x_1, \ldots, x_r)$, where $q(0, \ldots, 0, x_r) = \bar{q}(x_r)$ and $h(0, \ldots, 0, x_r) = x_r^m$.

We know that $q(0, \ldots, 0) \neq 0$ in N_1, thus we can say that there exists an $\epsilon > 0$ and $N_2 \subset N_1$, such that $q(0, \ldots, x_r) \neq 0$ for all $(x_1, \ldots, x_r) \in N_2 \times (-\epsilon, \epsilon)$. Since θ is a continuous map, we can shrink N_2 to get $\theta(x_1, \ldots, x_{r-1}) \in (-\epsilon, \epsilon)$ for all $(x_1, \ldots, x_{r-1}) \in N_2$. In essence the subsection of σ (of f over N_2) lies in $N_2 \times (-\epsilon, \epsilon)$. Let υ' be the lex-least valuation of f at α (in this case, the origin). Let $\alpha' \in S \cap N_2$. Since f is Lazard delineable, $\upsilon_{(\alpha', \theta(\alpha'))}(f) = \upsilon'$.

We can see that the lex-least valuation of h at the origin is $(0, \ldots, m)$. Since $f = q \cdot h$ and f is Lazard delineable in $S \cap N_2$, this implies that $\upsilon_{(\alpha', \theta(\alpha'))}(h) = (0, \ldots, m)$.

Denote P as the resultant of h and g with respect to x_r. As in the proof of Theorem 2.2 in [7] we see that $R = Q \cdot P$ for some suitable formal power series Q in the first $r - 1$ variables.

Since h and g are zero at the origin, this implies that $P = 0$ at the origin. This further implies that the lex-least valuation of P at the origin is non-zero. We know that R is lex-least invariant in the region N_2, which implies that P is lex-least invariant in N_2 (by the product law).

Since $h = 0$ in $N_2 \times (-\epsilon, \epsilon)$ and P is lex-least invariant in N_2, this implies that $g = 0$ in $\sigma \cap \{N_2 \times (-\epsilon, \epsilon)\}$ which is a euclidean neighbourhood of σ. Thus g is sign-invariant in σ, which implies that it is sign-invariant in S.

Theorem 3. *Let A be a set of pairwise relatively prime irreducible polynomials in r variables $x_1, ..., x_r$ of positive degrees in x_r, where $r \geq 2$. Let E be a subset of A. Let S be a connected submanifold of $\mathbb{R}^{r-1}$. Suppose that each element of $PL_E(A)$ is lex-least invariant in S. Then each element of E either vanishes identically on S or is Lazard delineable on S. The sections over S of the elements of E which do not vanish identically on S are pairwise disjoint, each element of E is lex-least invariant in every such section, and each element of A not in E is sign-invariant in every such section.*

Proof. This follows directly from Theorem 1 and Theorem 2.

Theorem 3 does come with its flaws, similar to McCallum's single equational constraint [7]. The method cannot be used recursively as it will output a sign-invariant CAD.

5 Conclusion and Further Research

The main requirement for Theorem 3 to work is that the EC in the QFF must not contain curtains. The other limiting factor for this projection is that it cannot be used recursively because it outputs a sign-invariant CAD, rather than a lex-least invariant CAD. This results in using the modified projection operator $PL_E(A)$ only in the first step of the projection phase, and Lazard's projection operator $PL(A)$ for the subsequent steps. Despite these drawbacks, reducing the output of the first step in the projection phase has a domino effect on further steps. McCallum's projection operator in [8] shows that it is possible to benefit from having an EC in the QFF, and provides an order-invariant CAD. By contrast with the projection operator in [7] which only provides a sign-invariant CAD. In [8] the list of resultants computed by the projection operator become the ECs for the next step of the projection phase. Similarly, we hope to find a projection operator that outputs a lex-least invariant CAD so that it can be used recursively. This modification would reduce the complexity of the algorithm significantly.

References

1. Brown, C., El Kahoui, M., Novotni, D., Weber, A.: Algorithmic methods for investigating equilibria in epidemic modeling. J. Symbolic Comp. **41**, 1157–1173 (2006)
2. Collins, G.: Quantifier Elimination for Real Closed Fields by Cylindrical Algebraic Decomposition. In: Proceedings 2nd. GI Conference Automata Theory & Formal Languages. pp. 134–183 (1975)
3. England, M., Bradford, R., Davenport, J.: Cylindrical Algebraic Decomposition with Equational Constraints. In: Davenport, J., England, M., Griggo, A., Sturm, T., Tinelli, C. (eds.) Symbolic Computation and Satisfiability Checking. Journal of Symbolic Computation (to appear) (2019)
4. England, M., Bradford, R., Davenport, J.: Cylindrical algebraic decomposition with equational constraints. Journal of Symbolic Computation (07 2019). https://doi.org/10.1016/j.jsc.2019.07.019
5. Lazard, D.: An Improved Projection Operator for Cylindrical Algebraic Decomposition. In: Bajaj, C. (ed.) Proceedings Algebraic Geometry and its Applications: Collections of Papers from Shreeram S. Abhyankar's 60th Birthday Conference. pp. 467–476 (1994)
6. McCallum, S.: An Improved Projection Operation for Cylindrical Algebraic Decomposition. Ph.D. thesis, University of Wisconsin-Madison Computer Science (1984)
7. McCallum, S.: On Projection in CAD-Based Quantifier Elimination with Equational Constraints. In: Dooley, S. (ed.) Proceedings ISSAC '99. pp. 145–149 (1999)
8. McCallum, S.: On Propagation of Equational Constraints in CAD-Based Quantifier Elimination. In: Mourrain, B. (ed.) Proceedings ISSAC 2001. pp. 223–230 (2001)
9. McCallum, S., Hong, H.: On Lazard's Valuation and CAD Construction. `http://www.arxiv.org/abs/1501.06563` (2015)
10. McCallum, S., Parusiński, A., Paunescu, L.: Validity proof of Lazard's method for CAD construction. J. Symbolic Comp. **92**, 52–69 (2019)
11. Mulligan, C., Davenport, J., England, M.: TheoryGuru: A Mathematica Package to Apply Quantifier Elimination Technology to Economics. In: Davenport, J., Kauers, M., Labahn, G., Urban, J. (eds.) Proceedings Mathematical Software — ICMS 2018. pp. 369–378 (2018)
12. Wada, Y., Matsuzaki, T., Terui, A., Arai, N.: An Automated Deduction and Its Implementation for Solving Problem of Sequence at University Entrance Examination. In: Proceedings ICMS 2016. pp. 82–92 (2016)